Quantum Property Testing for Solvable Groups

Yoshifumi Inui*
Department of Computer Science, The University of Tokyo
ERATO-SORST Quantum Computation and Information Project, JST

Abstract

Property testing has been extensively studied and its target is to determine whether a given object satisfies a certain property or it is far from the property. In this paper, we construct an efficient quantum algorithm which tests if a given quantum oracle performs the group multiplication of a solvable group. Our work is strongly based on the efficient classical testing algorithm for Abelian groups proposed by Friedl, Ivanyos and Santha. Since every Abelian group is a solvable group, our result is in a sense a generalization of their result.

1 Introduction

In order to guarantee the behavior of a program, we must prove it mathematically. For an ideal algorithm, constructing a perfect proof can be done, but for practical software, it is usually difficult and tedious. Furthermore such an approach cannot detect hardware based errors (common in the quantum setting). Testing the outputs through all inputs ensures the correctness of an algorithm exactly but it takes too much time. In this context, program checking [7, 8], self-testing [9] and self-correcting [9, 24] were introduced. A program checker checks if a program is correct on a particular input. A self-tester tests if a program is correct on most inputs. A self-corrector computes a correct value using a program which is correct on most inputs. In the study of these methods, the relation between a function f itself and a program P expected to compute f matters.

Property testing is another checking technique. It deals with efficient algorithms deciding whether a given object has some expected property or it is far from any object having that property. In testing a function, the difference between property testing and self-testing is that property tester tests only if a program has the property which the function has, and does not test if a program's behavior is the function itself. One of the interests of property testing is that, by running property testing algorithm in advance self-testing tasks become quite easy in many cases. The notion of property testing was first introduced

*psi@is.s.u-tokyo.ac.jp

by Rubinfeld and Sudan [32]. In this setting many properties including algebraic function properties [32, 12, 6, 18], graph properties [20, 2], computational geometry properties [1, 11] and regular languages [3, 16, 15] were proved to be testable. Quantum testers have also been studied and they are known to be more powerful than classical testers [10, 19, 26]. See [14, 30] for surveys on property testing.

The most active research areas in quantum computing are group theoretic problems. Shor's factorization algorithm [33] can be regarded as an instance of the hidden subgroup problem (HSP) (see [25] for a survey). The HSP over dihedral groups and symmetric groups are interesting and challenging because the former corresponds to the unique shortest vector problem [29] and the latter to the graph isomorphism problem. Other quantum algorithms for group theoretic problems [34, 13] are also known. In most of these problems, group operations are carried out using group oracles. Given two elements a, b in a group, the group oracle returns $a \cdot b$, and it returns the inverse a^{-1} given an element a. However its structure is unknown to us in the setting. Thus a natural question arises: is a given group oracle really what we expect?

Now our concern is to test if a given oracle has the property as a supposed group oracle. This test is related to the tests to decide whether a given oracle is that of some group and whether a function from a group to another group is a homomorphism. The second test has been studied [9, 6] and it can be done efficiently in the classical setting. About the first test, classical algorithms testing associativity of given functions were constructed in [12, 28]. These algorithms can be used for general groups, but their running times are polynomial in the size of the ground sets. As each elements in a set Γ is encoded in $O(\log |\Gamma|)$ bits, time complexities are exponential of the input lengths. However, Friedl, Ivanyos, and Santha have succeeded in designing a polynomial time algorithm which tests whether a given oracle is an Abelian group oracle or not [18]. Their testing algorithm is a classical one, but its basic idea came from the quantum setting.

In this paper, we construct an efficient quantum algorithm for the problem of testing whether a given oracle is that of a solvable group. The definition of solvable group is as follows.

Definition 1. *A group G is solvable if it has a normal subgroup chain $\{1\} = G_0 \lhd \cdots \lhd G_t = G$ such that G_{i+1}/G_i is an Abelian group for all i.*

It is obvious that an Abelian group is a solvable group. We can find much more solvable groups in quantum computation areas. Dihedral groups over which the HSP is solvable in subexponential time [23] are solvable groups. Moreover some non-Abelian groups over which the HSP are efficiently solvable are solvable [31, 21, 17, 27, 22, 5]. Watrous's algorithm [34] and Fenner and Zhang's algorithm [13] are also for solvable groups. As can be seen from the above examples, quantum oracles for solvable groups are used in many important algorithms. Hence it is natural to focus on the problem of testing if given oracles are really solvable group oracles.

2 Definitions

We give a formal definition for property testers.

Definition 2. *Let f be a function, P a set of functions, and d a distance function between functions. A quantum ϵ-tester for P with distance d is a quantum oracle Turing machine M such that*

$$\begin{cases} \mathbf{Pr}[M^f(\epsilon)\, accepts] > \frac{2}{3} & if\ d(f,P) = 0 \\ \mathbf{Pr}[M^f(\epsilon)\, accepts] < \frac{1}{3} & if\ d(f,P) > \epsilon. \end{cases}$$

We use $d(f,P)$ to represent $\inf_{g\in P} d(f,g)$ here.

Just like the Abelian case, in this paper we adopt the edit distance as the distance function, which we now define. The complete definition is described in [18]. A table is a square matrix and its size is the number of elements. We consider three operations to transform a table to another.

An exchange operation replaces elements in a table by arbitrary elements and its cost is a number of replaced elements. An insert operation inserts new rows and columns. A delete operation deletes existing rows and columns. The indices of inserted/deleted rows and columns must be equivalent. Its cost is a number of inserted/deleted elements.

We define edit distance for tables using the idea of transform, and extend it to magmas. A magma M is a set equipped with a binary operation $\cdot : M\times M \to M$. A multiplication table of a magma M is a table of which both rows and columns have a one-to-one correspondence with elements in M, $\phi : \{1,\cdots,|M|\} \to M$, and elements in the i-th row and the j-th column is equivalent to $\phi(i)\cdot\phi(j)$.

Definition 3. *Edit distance between two tables T and T' is the minimum cost needed to transform T to T' divided by the maximum size between T and T'.*

Edit distance between two magmas M and M' is the minimum edit distance between T and T' where $T[T']$ runs over all tables which is a multiplication table of $M[M']$. We identify two multiplication tables if we can obtain one from another only by renaming elements in its ground sets.

We say that a magma M is ϵ-close to another magma M' if the distance between them is less than or equal to ϵ, and that M is ϵ-far from M' otherwise.

3 Algorithm

3.1 Overview

We are given a set Γ and a quantum oracle which performs a binary operation $\cdot$ over Γ. This quantum oracle carries out the following unitary operations: $|a\rangle|b\rangle \mapsto |a\rangle|a\cdot b\rangle$ for all $a, b \in \Gamma$.

Theorem 4. *There exists a quantum ϵ-tester for solvable groups which runs in polynomial time in $\log|\Gamma|$ and ϵ^{-1}.*

Here, we identify a magma with its multiplication function.

We now overview our testing algorithm. The details are given in subsequent subsections.

First, we pick sufficiently many random elements $\alpha_1, \cdots, \alpha_s$ from the ground set Γ. As in Watrous's algorithm [34], we apply Babai et al.'s polynomial time Monte Carlo algorithm [4] which tests whether an input group is solvable. Notice that this algorithm requires that the input is a group, so we cannot decide if $(\Gamma, \cdot)$ is a solvable group at this stage. When the input is a group, we obtain the elements $h_1, \cdots, h_t$ satisfying

$$\{1\} = G_0 \lhd \cdots \lhd G_i = \langle h_1, \cdots, h_i \rangle \lhd \cdots \lhd G_t = \langle \alpha_1, \cdots, \alpha_s \rangle.$$

Now we define the set $H_j \subseteq \Gamma$ corresponding to G_j, as the following:

$$H_0 = \{1\}, H_j = \{h_j^a \cdot h | a \in \mathbb{Z}_{m_j}, h \in H_{j-1}\}.$$

The integer m_j is the order of h_j with respect to H_{j-1}, and the way to obtain it is given later.

Since H_0 is an identity group, it is obviously a solvable group. If we can test whether H_j is a solvable group under the assumption that H_{j-1} is a solvable group, we can test whether H_t is a solvable group inductively. We show how to test it.

Assume that H_{j-1} is $\frac{\epsilon}{2}$-close to a solvable group $\tilde{H}_{j-1}$, multiplication of which can be computed efficiently. In order to test if H_j is a solvable group, we construct a solvable group which is $\frac{\epsilon}{2}$-close to H_j using $\tilde{H}_{j-1}$. For this purpose we use the following theorem also used for Abelian groups.

Theorem 5 (Theorem 2 of [18]). *Let G be a group, $f : G \to H$ be a function and $\eta < \frac{1}{120}$. Assume that the inequality $\mathbf{Pr}_{x,y \in G}[f(xy) = f(x) \cdot f(y)] > 1 - \eta$ holds. Then there exists a group $\tilde{H}$ with multiplication $*$ and a homomorphism $\tilde{f} : G \to \tilde{H}$ such that*

1. $|\tilde{H} \backslash H| \leq 30\eta |\tilde{H}|$,
2. $\mathbf{Pr}_{\alpha,\beta \in \tilde{H}}[\alpha * \beta \neq \alpha \cdot \beta] \leq 91\eta$,
3. $\mathbf{Pr}_{x \in G}[\tilde{f}(x) \neq f(x)] \leq 30\eta$.

We construct a group G and a function f in such a way that $\tilde{f}$ is an isomorphism and $\tilde{H}$ is ϵ-close to H. And then for a random pair of x and y we test whether it satisfies $f(xy) = f(x) \cdot f(y)$, which we call a homomorphism test. We evaluate the probability inequality by conducting homomorphism tests several times. Applying the theorem, if the inequality is satisfied there exists a solvable group $\tilde{H}_j$ which is $\frac{\epsilon}{2}$-close to H_j.

As a result, we can test whether H_t is a solvable group. However there might exist elements in $\Gamma \backslash H_t$. In order to guarantee the fraction of such elements is less than $\frac{\epsilon}{2}$, we pick $O(\epsilon^{-1})$ random elements from Γ and decompose them over H_t. If there exist elements that cannot be decomposed, we reject.

We summarize the algorithm briefly.

1. Find generators of a normal subgroup chain.
2. For each $1 \leq j \leq t$ do (a) and (b).
   (a) Find the order of h_j w.r.t. H_{j-1}.
   (b) Test if f is a homomorphism.
3. Test if there exist elements in $\Gamma \backslash H_t$.

The rest of this chapter covers the shortcoming of details and has the following organization. Subsection 3.2 deals with the problem of finding the order of h_j with respect to H_{j-1}, which is the key task in Watrous's order finding algorithm for solvable groups [34]. In subsection 3.3, we show the algorithm to decompose an element over H_j. Utilizing the results obtained in these two subsections, in subsection 3.4 we construct a solvable group G and a function f in Theorem 5. With these G and f we apply Theorem 5 and show that the group $\tilde{H}_j$ is $\frac{\epsilon}{2}$-close to H_j in subsection 3.5. Correctness and complexity of the algorithm are discussed in subsection 3.6.

3.2 The order of h_j with respect to H_{j-1}

When H_j is a solvable group, we call $m_j := \min\{m | h_j^m \in H_{j-1}\}$ the order of h_j with respect to H_{j-1}, which can be gained using Watrous's algorithm [34]. Also in a non-group case, we call the value obtained from Watrous's algorithm the order of h_j with respect to H_{j-1}. The details of the algorithm are as follows.

We prepare the superposition state over $\tilde{H}_{j-1}$, $|\tilde{H}_{j-1}\rangle = \frac{1}{\sqrt{|\tilde{H}_{j-1}|}} \sum_{h \in \tilde{H}_{j-1}} |h\rangle$, and the state over $\mathbb{Z}_r$, $\frac{1}{\sqrt{r}} \sum_{a=0}^{r-1} |a\rangle$ where r is the order of h_j and can be found by Shor's order finding algorithm [33]. Then we apply the following operations:

$$\frac{1}{\sqrt{r}} \sum_{a=0}^{r-1} |a\rangle |\tilde{H}_{j-1}\rangle \mapsto \frac{1}{\sqrt{r}} \sum_{a=0}^{r-1} |a\rangle |h_j^a \cdot \tilde{H}_{j-1}\rangle \mapsto \frac{1}{r} \sum_{a,b=0}^{r-1} e^{2\pi i \frac{ab}{r}} |b\rangle |h_j^a \cdot \tilde{H}_{j-1}\rangle.$$

The latter operation is the quantum Fourier transform modulo r. The multiplication between h_j^a and $\tilde{h} \in \tilde{H}_{j-1}$ is $h_j^a \cdot h$ where h is an element in H_{j-1} corresponding to $\tilde{h}$. By observing the first register, we obtain $\tilde{b}$ which is a multiple of $\frac{r}{m_j}$ since $|h_j^{m_j} \cdot \tilde{H}_{j-1}\rangle = |\tilde{H}_{j-1}\rangle$. Repetition yields the desired value m_j. For such m_j we check $h_j^{m_j} \in H_{j-1}$ by decomposing $h_j^{m_j}$.

3.3 Decomposition over H_j

Testing if an element h in H_j is really a member of H_j reduces to the task of decomposing h over H_j. The decomposition algorithm is as follows.

Suppose $h = h_{j-1}^{a_{j-1}} \cdots h_1^{a_1}$. First we prepare the following state:

$$\sum_{a,b} |a\rangle |b\rangle |h^a \cdot (h_j^b \cdot \tilde{H}_{j-1})\rangle.$$

We can obtain a_j as we do for discrete logarithm problem, because $|h^a \cdot (h_j^b \cdot \tilde{H}_{j-1})\rangle = |\tilde{H}_{j-1}\rangle$ for a and b satisfying $h^a \cdot h_j^b \in \tilde{H}_{j-1}$, that is $aa_j + b = 0$. Next we do similarly for $h \cdot h_j^{-a_j}$. Thus we can find each a_k inductively, and we obtain a decomposition of h finally.

3.4 Construction of G and f

We construct an appropriate solvable group G and a function f in Theorem 5 and claim that $\tilde{H}_j$ is ϵ-close to H_j for such G and f.

We define the group G first. We extend $\tilde{H}_{j-1}$ which is almost isomorphic to $\tilde{H}_{j-1}$ to a solvable group $\tilde{H}_j$ by adding h_j so that $\tilde{H}_j$ is almost isomorphic to H_j. Consider a pair of two groups $\mathbb{Z}_{m_j} \times \tilde{H}_{j-1}$ and introduce a multiplication $\circ$ over it satisfying $(m, h) \circ (n, h') = (m + n, \phi^n(h)h')$. Here, ϕ satisfies the following conditions:

$$\begin{cases} \phi(hh') &= \phi(h)\phi(h') \\ \phi(h_i) &= h_j^{-1} \cdot (h_i \cdot h_j). \end{cases}$$

If ϕ is an automorhphism of H_{j-1}, $\mathbb{Z}_{m_j} \times \tilde{H}_{j-1}$ is a solvable group with the multiplication $\circ$, because it satisfies the definition of a group and $(0, \tilde{H}_{j-1})$ is its normal subgroup. In order to check if ϕ is an automorphism, we must check $\forall i\ \phi(h_i) \in \tilde{H}_{j-1}$ and $|H_{j-1}| = |\phi(H_{j-1})|$. The former can be done by decomposing $\phi(h_i)$ over $\tilde{H}_{j-1}$ and the latter by Watrous's algorithm.

We have defined the group G, so next define the function ψ like this:

$$\begin{array}{rcl} \psi : \mathbb{Z}_{m_j} \times \tilde{H}_{j-1} & \to & H_j \\ (n, \tilde{h}) & \mapsto & h_j^n \cdot h. \end{array}$$

Here $\tilde{h}$ is represented as $(a_{j-1}, \cdots, a_1)$ and h is defined as $h_{j-1}^{a_{j-1}} \cdot \cdots \cdot h_1^{a_1}$ for such a_is.

3.5 Application of Theorem 5

In order to conduct a homomorphism test with $G = (\mathbb{Z}_{m_j} \times \tilde{H}_{j-1}, \circ)$ and $f = \psi$ which we have constructed in the previous subsection, we must compute $\psi(xy)$ for $x, y \in \tilde{H}_j$ efficiently. To that end, we need to know the value $\phi^n(h)$ for $h \in \tilde{H}_{j-1}$. We show how to obtain it. Let h be represented as $(a_{j-1}, \cdots, a_1)$. Because

$$\phi^n({h_{j-1}}^{a_{j-1}} \cdots h_1^{a_1}) = \phi^n(h_{j-1})^{a_{j-1}} \cdots \phi^n(h_1)^{a_1}$$

holds, we only need the value $\phi^n(h_i)$ for each i and n. We already know the value $\phi(h_i)$ by decomposition. Assume we know the value $\phi^{2^k}(h_i)$ for each i. The value $\phi^{2^{k+1}}(h_i)$ is equivalent to $\phi^{2^k}(\phi^{2^k}(h_i))$, hence it can also be computed efficiently. The value for n which is not a power of two can be computed similarly. Therefore we can obtain all values inductively and compute the multiplication over G efficiently.

Now by applying the theorem we obtain the homomorphism $\tilde{\psi}$ and the group $\tilde{H}_j$, which we want to conclude is $\frac{\epsilon}{2}$-close to H_j.

Proposition 6. *Assume H_{j-1} is $\frac{\epsilon}{2}$-close to a solvable group $\tilde{H}_{j-1}$. If ψ constructed in the previous subsection satisfies $\mathbf{Pr}_{x,y\in G}[\psi(x \circ y) = \psi(x) \cdot \psi(y)] > 1 - \eta$, the group $\tilde{H}_j$ the existence of which is guaranteed by Theorem 5 is 151η-close to H_j.*

Proof. The distance between $\tilde{H}_j$ and H_j is determined by the number of elements being a member of either set and the number of pairs of two elements multiplication of which differ by operation. In fact, to transform a multiplication table of $\tilde{H}_j$ to that of H_j, we first delete and insert rows and columns corresponding to elements in $\tilde{H}_j \backslash H_j$ and $H_j \backslash \tilde{H}_j$ respectively, and then exchange multiplication values which differ between two tables. It follows that the number of elements in $\tilde{H}_j \backslash H_j$ and the number of pairs, γ_1 and γ_2 such that $\gamma_1 \circ \gamma_2 \neq \gamma_1 \cdot \gamma_2$, are small from 1 and 2 in Theorem 5. To complete the proof, we must show that the size of $H_j \backslash \tilde{H}_j$ is small. This follows from the theorem if $|\tilde{H}_j|$ is greater than or equal to $|H_j|$.

As $\tilde{\psi}$ is a homomorphism from G, $\tilde{H}_j$ is isomorphic to a subgroup of G. Note that $\tilde{H}_j$ cannot be isomorphic to a group of the form $(\mathbb{Z}_{m_j} \times H', \circ)$ where $H' \lneq \tilde{H}_{j-1}$, because the contradiction that unitary operation $|h_j^n\rangle|h\rangle \mapsto |h_j^n\rangle|h_j^n \cdot h\rangle$ is not invertible follows from 3 of Theorem 5. Suppose $\langle(n_0, \tilde{h}_0)\rangle$ is the kernel of $\tilde{\psi}$. Since it is a subgroup of G, it follows that n_0 is either m_j or a strict divisor of m_j.

Assume $n_0 \neq m_j$ holds for contradiction. Recall how we found m_j. The theorem tell us for almost all n and $\tilde{h}$ the value $h_j^n \cdot h$ is same with $\tilde{\psi}(n, \tilde{h})$. Therefore we must have obtained n_0 instead of m_j since $|\tilde{h}_j^{n_0} \cdot \tilde{H}_{j-1}\rangle = |\tilde{H}_{j-1}\rangle$. This is a contradiction. It follows that $n_0 = m_j$, therefore $G \cong \tilde{H}_j$ holds. It indicates that $|\tilde{H}_j| = |G| = m_j|\tilde{H}_{j-1}| \geq |H_j|$. □

By taking η so that $151\eta \leq \frac{\epsilon}{2}$, we can test whether H_j is $\frac{\epsilon}{2}$-close to a solvable group under the assumption that H_{j-1} is $\frac{\epsilon}{2}$-close to a solvable group $\tilde{H}_{j-1}$.

3.6 Analysis

In this subsection, we discuss the correctness and the complexity of the algorithm. We run the above algorithm to test whether a given oracle is that of a solvable group or not. If an error occurs during the execution we reject, otherwise we accept. We expect the algorithm to accept a solvable group oracle and reject the others with high probability, and work in polynomial time in $\log|\Gamma|$ and ϵ^{-1}.

It is obvious that a solvable group oracle passes the test with high probability. We would like to show that the oracle ϵ-far from that of a solvable group is rejected with high probability. Take an oracle ϵ-far from that of any solvable group, then H_t is $\frac{\epsilon}{2}$-far from $\tilde{H}_t$ or $|\Gamma \backslash H_t| > \frac{\epsilon}{2}$ holds. If the latter holds, it

should be rejected with high probability; hence we consider the former case. Suppose that the oracle passes all homomorphism tests. From Proposition 6, since it passes the homomorphism tests the distance between H_{t-1} and any solvable group $\tilde{H}_{t-1}$ is at least $\frac{\epsilon}{2}$. By repeating the above arguments, we see that the cyclic group H_1 is $\frac{\epsilon}{2}$-far from any cyclic group but it passes the test for Abelian groups. This is a contradiction.

We turn to the time complexity of the algorithm. The number of iterations of the task 3 in the diagram in subsection 3.1, t, is polynomial because the output size of the task 1 is also polynomial. We can easily know that the computation time of each task other than 3-(b) is polynomial. How many times do we need to repeat the homomorphism tests? Theorem 5 and Proposition 6 tell us that suppose the stage $j = k$ is the first stage such that the distance between H_k and $\tilde{H}_k$ is greater than ϵ we fail in the homomorphism test with probability greater than η. When we repeat the tests c times, the probability never to fail is less than $(1-\eta)^c$. We are able to take $c = O(\frac{1}{\eta})$ to bound the success probability by $\frac{1}{3}$. As η is taken to be a constant times ϵ, the total computation time is polynomial in $\log \Gamma$ and ϵ^{-1} as we stated.

References

[1] N. Alon, S. Dar, M. Parnas and D. Ron, *Testing of clustering*, Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science, 240–250, 2000.

[2] N. Alon, E. Fischer, I. Newman and A. Shapira, *A combinatorial characterization of the testable graph properties: It's all about regularity*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 251–260, 2006.

[3] N. Alon, M. Krivelevich, I. Newman and M. Szegedy, *Regular languages are testable with a constant number of queries*, Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science, 656–666, 1999.

[4] L. Babai, G. Cooperman, L. Finkelstein, E. Luks and Á. Seress, *Fast Monte Carlo algorithms for permutation groups*, Journal of Computer and System Sciences, 50:296–307, 1995.

[5] D. Bacon, A. Childs and W. van Dam. *From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups* Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, 469–478, 2005.

[6] M. Ben-Or, D. Coppersmith, M. Luby and R. Rubinfeld, *Non-Abelian homomorphism testing, and distributions close to their self-convolutions*, Proceedings of the 8th International Workshop on Randomization and Computation, 273–285, 2004.

[7] M. Blum, *Designing programs to check their work*, Technical Report TR-88-009, International Computer Science Institute, 1988.

[8] M. Blum and S. Kanna, *Designing programs that check their work*, Proceedings of the 21st Annual ACM Symposium on Theory of Computing, 86–97, 1989.

[9] M. Blum, M. Luby and R. Rubinfeld, *Self-testing/correcting with applications to numerical problems*, Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, 73–83, 1990.

[10] H. Buhrman, L. Fortnow, I. Newman and H. Röhrig, *Quantum property testing*, Proceedings of the 14th annual ACM-SIAM Symposium on Discrete algorithms, 873–882, 2001.

[11] A. Czumaj and C. Sohler, *Property testing in computational geometry*, Proceedings of the 8th Annual European Symposium on Algorithms, 155-166, 2000.

[12] F. Ergün, S. Kannan, R. Kumar, R. Rubinfeld and M. Viswanathan, *Spot-checkers*, Journal of Computer and System Sciences, 60(3):717–751, 2000.

[13] S. Fenner and Y. Zhang, *Quantum algorithms for a set of group theoretic problems*, Proceedings of the 9th Italian Conference on Theoretical Computer Science, Lecture Notes in Computer Science, 3701:215–227, 2005.

[14] E. Fischer, *The art of uninformed decisions: A primer to property testing*, The Computational Complexity Column of The Bulletin of the European Association for Theoretical Computer Science, 75:97–126, 2001.

[15] E. Fischer, G. Kindler, D. Ron, S. Safra and A. Samorodnitsky, *Testing juntas*, Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, 103–112, 2002.

[16] E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld and A. Samorodnitsky, *Monotonicity testing over general poset domains*, Proceedings of the 34th Annual ACM Symposium on Theory of Computing, 474–483, 2002.

[17] K. Friedl, G. Ivanyos, F. Magniez, M. Santha and P. Sen. *Hidden translation and orbit coset in quantum computing*, Proceedings of the 35th Annual ACM Symposium on Theory of Computing, 1–9, 2003.

[18] K. Friedl, G. Ivanyos and M. Santha, *Efficient testing of groups*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 157–166, 2005.

[19] K. Friedl, F. Magniez, M. Santha and P. Sen, *Quantum testers for hidden group properties*, Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, 2747:419–428, 2003.

[20] O. Goldreich, S. Goldwasser and D. Ron, *Property testing and its connection to learning and approximation*, Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 339-348, 1996.

[21] S. Hallgren, A. Russel and A. Ta-Shma. *Normal subgroup reconstruction and quantum computing using group representations*, Proceedings of the 32th Annual ACM Symposium on Theory of Computing, 627–635, 2000.

[22] Y. Inui and F. Le Gall. *An efficient algorithm for the hidden subgroup problem over a class of semi-direct product groups*, arXiv: quant-ph/0412033, 2004.

[23] G. Kuperberg, *A subexponential-time quantum algorithm for the dihedral hidden subgroup problem*, SIAM Journal on Computing, 35(1):170–188, 2005.

[24] R. Lipton, *New directions in testing*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 2:191–202, 1991.

[25] C. Lomont *The hidden subgroup problem - Review and open problems*, arXiv: quant-ph/0411037, 2004.

[26] F. Magniez and A. Nayak, *Quantum complexity of testing group commutativity*, Proceedings of 32nd International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, 1770:1312–1324, 2005.

[27] C. Moore, D. Rockmore, A. Russell and L. Schulman. *The power of basis selection in Fourier sampling: Hidden subgroup problems in affine groups*, Proceedings of the 15th Annual ACM Symposium on Discrete Algorithms, 1106–1115, 2004.

[28] S. Rajagopalan and L. Schulman, *Verification of identities*, Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 612–616, 1996.

[29] O. Regev, *Quantum computation and lattice problems*, SIAM Journal on Computing, 33(3), 738–760, 2004.

[30] D. Ron, *Property testing*, In Handbook of Randomized Computing, Kluwer Academic Publishers, 597- -649, 2001.

[31] M. Rötteler and T. Beth. *Polynomial-time solution to the hidden subgroup problem for a class of non-Abelian groups*, arXiv: quant-ph/9812070, 1998.

[32] R. Rubinfeld and M. Sudan, *Robust characterizations of polynomials with applications to program testing*, SIAM Journal of Computing, 25(2), 252–271, 1996.

[33] P. Shor, *Polynomial-Time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Journal on Computing, 26(5), 1484–1509, 1997.

[34] J. Watrous, *Quantum algorithms for solvable groups*, Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 60–67, 2001.